In the Specification:

The paragraph on page 1, spanning lines 6 through 10, has been amended as shown below:

The present application is related to U.S. Patent Application No. 09/599,015, entitled FILTERING A PERMISSION SET USING PERMISSION REQUESTS ASSOCIATED WITH A CODE ASSEMBLY, and U.S. Patent Application No. 09/598,814, entitled EVALUATING INITIALLY UNTRUSTED EVIDENCE IN AN EVIDENCE-BASED SECURITY POLICY MANAGER, filed concurrently herewith and assigned to the Assignee of the present invention.

The paragraph spanning page 4, line 22, through page 5, line 15, has been amended as shown below:

In other implementations of the present invention, articles of manufacture are provided as computer program products. One embodiment of a computer program product provides a computer program storage medium readable by a computer system and encoding a computer program for executing a computer process for generating a permission grant set for a code assembly received from a resource location. Another embodiment of a computer program product may be provided in computer data signal embodied in a carrier wave by a computing system and encoding the computer program for generating a permission grant set for a code assembly received from a resource location. The computer program product encodes a computer program for executing on a computer system a computer process for generating a permission grant set for a code assembly received from a resource location ~~is provided~~. The code assembly is associated with an evidence set. A security policy specification defining at least one code group collection having a plurality of code groups is received. Each code group is associated with a code-group permission set. The evidence set is evaluated relative to the code group collection to determine membership of the code assembly in one or more code groups of the code group collection. The permission grant set is generated based on one or more code-group permission sets, such that each code-group permission set is associated with a code group in which the code assembly is a member.

The paragraph on page 11, spanning lines 14 through 18, has been amended as shown below:

For example, the virtual machine executes a first code assembly (e.g., main code assembly 202 of FIG. 2) that calls a routine provided by a second code assembly (e.g., parser code assembly 204 of FIG. 2). The class loader 113 receives the virtual machine's request for the second code assembly and loads the second code assembly into the run-time call stack 114 so that the first code assembly can call the needed routine.

The paragraph on page 31, spanning lines 3 through 9, has been amended as shown below:

Each code assembly is evaluated against the membership criterion of each child of the "All" code group, which are specified by the child definition of the "All" code group. As illustrated, the "All" code group in FIG. 6 has four child code groups: "Publisher:Microsoft" 602, "Zone:Internet" 604, "Zone:Intranet" 606, and "Site:*.xyz.com" 608. The code group 602 has two child code groups: "Name:MS.Office" 610 and "Name:MS.Windows" 612. The code group 606 has two child code groups: "Publisher:CorpAdmin" 614 and "Site:*.LocalWeb.com" 616.

The paragraph spanning page 32, line 20, to page 33, line 6, has been amended as shown below:

FIG. 7 illustrates an exemplary system useful for implementing an embodiment of the present invention. An exemplary computing system for embodiments of the invention includes a general purpose computing device in the form of a conventional computer system 700, including a processor unit 702, a system memory 704, and a system bus 706 that couples various system components including the system memory 704 to the processor unit ~~700~~ 702. The system bus 706 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus and a local bus using any of a variety of bus architectures. The system memory includes read only memory (ROM) 708 and random access memory (RAM) 710. A basic input/output system ~~712~~ 711 (BIOS), which contains basic routines that help transfer information between elements within the computer system 700, is stored in ROM 708.

The paragraph on page 37, spanning lines 7 through 13, has been amended as shown below:

The decision operation 918 directs operation flow to a traversing operation 916 if the traversal queue is not empty. The traversal operation 916 traverses to the next code group in the traversal queue. Alternatively, the decision operation 916 directs operation flow to a merging operation 920 if the traversal queue is empty. The ~~emerging~~ merging operation 920 computes the union of the recorded code-group permission sets in an embodiment of the present invention. The permission grant operation 922 then provides the result of the merging operation 920 as the permission grant set.

The Abstract, spanning lines 3 through 16 of page 57, has been amended as shown below:

An evidence-based policy manager generates a permission grant set for a code assembly received from a resource location. The policy manager executes in a computer system (e.g., a Web client or server) in combination with the verification module and class loader of the run-time environment. The permission grant set generated for a code assembly is applied in the run-time call stack to help the system determine whether a given system operation by the code assembly is authorized. Both code assemblies and evidence may be received from a local origin or from a remote resource location via a network (e.g., the Internet). The policy manager may comprise execution modules for parsing a security policy specification, generating a one or more code hierarchies, evaluating membership of the received code assembly in one or more code groups, and generating a permission grant set based upon this membership evaluation. ~~The policy manager may generate multiple policy levels in accordance with a security policy definition specified in a security policy specification. Permission sets from each policy level may be merged to generate a permission grant set associated with the code assembly and applied in the run-time call stack of the execution thread.~~

In the Drawing:

Fig. 7 has been amended as shown in the marked-up-in-red copy thereof enclosed herewith, to correct a minor typographical error.